Fachhochschule
Nordwestschweiz

Scraping with Selenium

Instagram Posts and their Comments

Authors: **Bruno Baptista Kreiner, Katarina Fatur**

Mentor: **Fernando Benites**

Data Science BSc Study

Module: **Module Web Data Acquisition (*Web Datenbeschaffung*)**

Autumn Semester 2022

Motivation **1**
Why Scraping Instagram? 1
Project **1**
Architecture and Project Description 1
Data Analysis **2**
Dataset 2
Insights 2
Outlook **3**
Sources **3**

Motivation

Why Scraping Instagram?

While Twitter is a go-to place to measure the pulse on social exchange through text analysis, Instagram is primarily a visual platform, so from a technical point of view, it offers even more potential for analysis. In this scraping project we focused on the Instagram posts and their comments. If we later decide to extend the scraping to images and extract their features with machine-learning, it will be advantageous that we are already familiar with the inherent problems of Instagram scraping. The comments are less structured than Twitter replies, there are many images and videos, and a lot of sentiment is communicated through emojis. Along with the website's dynamic elements, notorious attempts at blocking scrapers and the fact that there is content that is available only for signed-in users, scraping Instagram seemed as an interesting task to do.

Introduction

We undertook this project as a partial fulfillment of the requirements to complete the WDB Module (web data acquisition). The task was to make a Selenium scraper and to analyze the scraped data.

To scrape an account we use Selenium, because it is able to circumvent the log-in and makes timing of the scraping-subtasks easier. Originally, Selenium was meant as a tool to test a website's behavior, but now it is widely used as an automation tool in web-scraping and other automation tasks for web browsers.

Due to the controversy surrounding Elon Musk, his unforeseen business decisions and sociopolitical comments, we were interested if we could get some insight into the extent of public interest in his posts. Since he doesn't have an official Instagram account, we chose a substitute fan account @elonmusk.

We wanted to obtain Musk's posts and related comments to assess if the public is responsive to Musk's actions. We have not had experience with analyzing Instagram content before and we were interested to work with emojis.

We scraped a fan-maintained account, but perusing a few comments we noticed that many people do not really discern whether an account is really maintained by Musk or not, they simply react to something he says or does in the real world.

Project

The project can be run inside a Docker or by creating a virtual environment manually (from the requirements.txt file). A concise summary of the project is provided below, but for the instruction on how to run it please refer to the README.md.

Architecture and Project Description

The project stores the data into the data folder. The scraped post data is stored in the *./data/tabular* folder. The scraper stores all post urls directly into the *./data* folder itself. The filenames of the post_urls and the scraped posts are the same for every scrape.
For each scrape the scraper goes through the same steps: scraping all post urls, storing those post urls in a text file inside *./data/* and then iterating through each url to get a post's data and store it into the database inside *./data/tabular/*.

For each post, the scraper first loads all the comments, then loads all the potential replies to each comment, and finally stores all the information mentioned in "Data Structure" into the database.

The scraping script accepts the following arguments:

- **account-name:** Required. The account name of the account to be scraped.
- **posts**: Defaults to None. If set to a valid txt file with one post url per line, the scraper will skip getting all posts of an account and try to scrape through each line in the file.
- **only-get-post-urls**: Defaults to False. If set to True, the script ends after storing all post urls and won't scrape through the posts itself.
- **from-post-url**: Defaults to None. If set to a valid post-url in the format 'https://www.instagram.com/p/[post id]/', the scraper will find the index of the post url in the posts' file and begin scraping posts from the index of the given post url.

Data Analysis

Automated extraction of data from social platforms like Instagram enables us to gain valuable insight into social trends, accounts engagement value, or current (social and political) trends. This information is not relevant only for research, but also for marketing and product design purposes, because it enables the stakeholders to understand their clients or audience better.

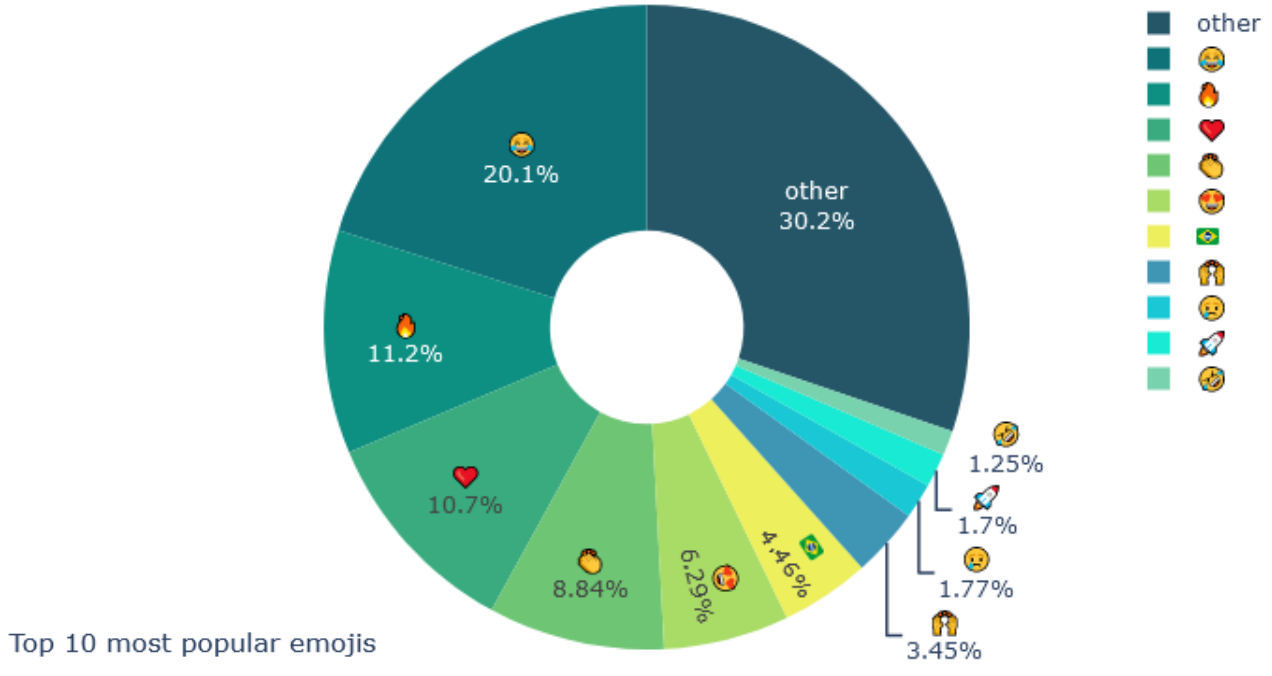
With libraries like emoji and nltk, the analysis possibilities extend beyond simple text. A while ago Airbnb asked their followers on Twitter to describe their dream trip in 3 emojis. What a clever idea for a mini survey about the Airbnb clients' interests!

In our analysis we focused on the most commonly used words and emojis. We also calculated the Instagram engagement rate, which was 2.5% (average - 1.5%; good - 3.6%). This is understandable, since the account is not the primary source of information for the true Elon Musk followers.

Dataset

We obtained 94,645 posts and comments (366 posts and almost 94,000 comments). We stored the information in 9 columns: *'post_url'*, *'is_post'*, *'commenter'*, *'text'*, *'replies_to'*, *'replies_count'*, *'likes'*, *'date'*, and *'crawl_time'*.

Insights



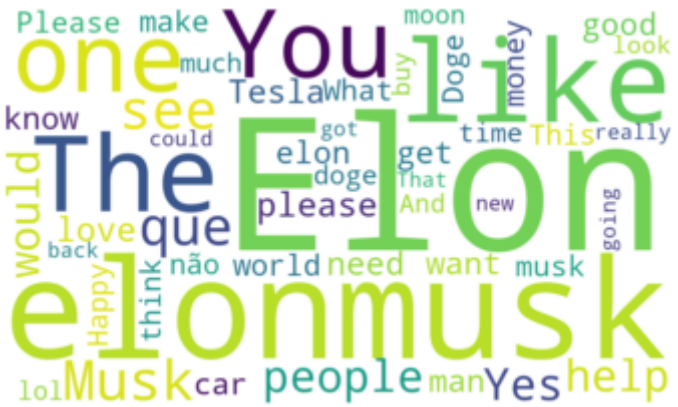
Top 10 most popular emojis

Emojis

Over 55% messages have no emojis, 26% of messages contain up to 4 emojis. *Face_with_tears_of_joy* is the prevalent emoji, and if we consider it together with its relatives *rolling_on_the_floor_laughing* and *face_with_tears_of_joy,* we can conclude that the main sentiment conveyed by emojis is rather positive. Unexpected was the *brazilian_flag* emoji, while the *rocket* emoji was somewhat underrepresented.

Text

The text on Instagram is secondary and thus mostly short. Distribution of word number per message is markedly right-skewed. While the maximum number of characters per message was 2272, the median number of characters was 21. In terms of words, this amounts to a maximum of 441 words per message, with the median number of words being just 4. The most frequently used words were 'Elon' and 'elonmusk', as evident from the word cloud.



Outlook

We could expand our tokenizing from simple word analysis to chunking the text into phrases and extract the most cliché expressions. Does the same user use a phrase often? Which phrases are catchy? Which phrases give rise to a new meme?

The word cloud above provides further ideas of what could be our next steps when processing the text with nltk. We notice the words "*Doge*", "*Tesla*", "*não*" and "*que*". Using Named Entity Recognition (NER) we could find the named entities in the texts and determine what kind of named entity they are (specific locations, people, organizations, etc.). Which products are most associated with the Elon Musk's brand? Where does the majority of commenters come from, Brazil?

We could also analyze the timing of the posts and comments, and check if the frequency of commenting follows the Poisson distribution. Is the rate λ in this case suggestive of the importance of a post?

This exploration would be even more exciting with an important business or political account. Thanks to our experimentation with Instagram scraping we now have the foundation for scraping such data.

Sources

What is Instagram Scraping?
Airbnb Experiences: An Emoji Story
Instagram Engagement Rate Calculator
Natural Language Processing With Python's NLTK Package